Einführung in Software Engineering WS 10/11

Fachbereich Informatik

Dr. Michael Eichberg
eichberg@informatik.tu-darmstadt.de

Assistent: Ralf Mitschke
mitschke@st.informatik.tu-darmstadt.de

TECHNISCHE UNIVERSITÄT DARMSTADT

Übungsblatt 7 (10 Punkte): Software Engineering Prozesse

Abgabeformat: Reichen Sie ihre Lösung per SVN ein. Jede Übung muss in einem eigenen Ordner ***ex<Number>*** *(<Number> = 01, 02, ...)* in Ihrem Gruppenverzeichnis eingereicht werden. Während der Übungsbearbeitung können Sie Ihre Lösungen beliebig oft in das SVN hochladen (per Commit). Wir prüfen die Zeit der Einreichung Ihrer Lösungen unter der Benutzung des SVN Zeitstempels.

Erstellen Sie für Lösungen der Aufgaben, die keinen Quelltext erfordern, eine PDF-Datei mit dem Dateinamen ***solution.pdf***. Dies gilt auch für alle UML-Diagramme, die Sie erstellen. Die Basisanwendung wird als Eclipse-Projekt vorgegeben. Ihr eigener Code muss entsprechend in den dafür vorgesehenen Verzeichnissen (***/src*** oder ***/test***) erstellt werden.

Abgabetermin: 12.01.2011 - 24:00 Uhr

Aufgabe 1 (3 Punkte)

Ziel: *Erfahrung mit der Planung von Softwareprojekten sammeln*

Im Folgenden wurden User-Storys für die Flashcards-Anwendung erstellt. Diese User-stories beinhalten eine Beschreibung, der zu implementierenden Funktionalität sowie eine Motivation, warum diese für den Kunden wichtig ist (Kursiv gestellt).

Überlegen Sie sich wie komplex die einzelnen Stories sind. Komplexität ist ein Maß für die Zeit, die Sie für eine Implementierung benötigen würden. Hier sollten Sie sowohl die Komplexität der Funktionalität, als auch Ihren Erfahrungsstand in Java einfließen lassen. Achten Sie auch darauf, dass zu jeder Story auch (JUnit-)Tests geschrieben werden müssen.

Vergeben Sie zur Schätzung Story-Points (wie in der Vorlesung zum Thema *Extreme Programming* vorgestellt) und Begründen Sie Ihre Schätzung kurz in einem Absatz.

User-Story 1

Die zum Erlernen neuer Inhalte aufgewendete Zeit ist ein wichtiger Faktor in der Bewertung des Lernerfolgs.

Die Flashcards-Anwendung soll während die Karte gelernt wird (Frageseite wird angezeigt) eine Stoppuhr anzeigen. Die Stoppuhr soll sekundengenau anzeigen, wie lange bisher auf der Frageseite verweilt wurde. Die Stoppuhr beginnt für jede Karte wieder bei null.

User-Story 2

Eine gute Übersicht über den aktuellen Lernerfolg ist sehr wichtig. Der Lernende soll möglichst schnell erkennen können, wo er schwächen hat. Eine farbliche Kennzeichnung des Lernerfolgs soll eine schnelle und vollständige Übersicht über alle Karteikarten bieten.

Die Flashcards-Anwendung soll detaillierte Statistiken zu jeder Karteikarte führen: Wann wurde die Karte gelernt; hat sich der Benutzer zu diesem Zeitpunkt korrekt an die Antwort erinnert; wie oft wurde die Karte gelernt; wie lange wurde die Karte gelernt und wie oft wurde die Karte hintereinander korrekt gelernt. Wir wünschen uns auch während des Lernens eine permanente, aktuelle Anzeige dieser Statistiken.

Die Statistiken sollen in eine Farbcodierung einfließen, die angibt wie häufig eine Karteikarte gewusst wurde (grün häufig; gelb selten; rot nie). Alle Statistikdaten müssen gespeichert werden und auf nicht gespeicherte Daten wird vor dem Beenden der Anwendung hingewiesen.

User-Story 3

Jeder lernt anders; Kinder lernen anders als Studenten oder Schüler. Für Matheformeln benötigt man möglicherweise länger als für Vokabeln oder umgekehrt. Die Flashcards-Anwendung soll das Lernen auf unterschiedliche Arten ermöglichen.

Es soll möglich sein ohne Systematik (durch herausziehen einer zufälligen Karte) zu lernen und das mit und ohne Zurücklegen der Karten. Sowohl nur die neuesten Karten, als auch ausschließlich ältere Karten sollen gezielt lernbar sein. Die Anzahl der Karten für eine Lernsitzung sollte man vorher einstellen können.

Besonders wichtig ist die Möglichkeit die Karten zum Lernen in fünf verschiedene Fächer einzuteilen. Dies ist der Standardfall der Domäne wenn mit physischen Karten gelernt wird. Neue Karten kommen in das vorderste Fach. Immer wenn man bei einer Karte erfolgreich war soll sie in das nächste Fach einsortiert werden. Innerhalb der Fächer sollen die Karten danach sortiert werden, wann man sich das letzte mal an die Antwort erinnert hat.

User-Story 4

Um die Übersichtlichkeit zu verbessern und den Zugriff auf gesuchte Flashkarten zu beschleunigen, ist eine Suchfunktion von Vorteil. Die Suche soll möglichst interaktiv sein. Die Übersichtlichkeit kann durch eine Sortierung der Flashcards weiter erhöht werden.

Die Flashcards-Anwendung soll eine Suchfunktion bieten, die vergleichbar ist mit denen der großen Suchanbieter (aus dem Internet) bzw. den integrierten Suchfunktionen der aktuellen Betriebssysteme. Sobald man etwas in das Suchfeld eintippt sollen nur noch Flashcards angezeigt werden deren Vorderseite dem Suchbegriff entsprechen.

Es soll dazu möglich sein die Flashkarten nach folgenden Kriterien zu sortieren: Erstellungsdatum (neueste zuerst), Datum des letzten erfolgreichen Lernens (ältestes Datum zuerst) und danach wie oft bereits eine Karte erfolgreich gelernt wurde (weniger häufig zuerst).

User-Story 5

Um den Bedienkomfort zu erhöhen und den Benutzern eine Menge Frust zu ersparen, sollen sämtliche Änderungen rückgängig gemacht werden können, die in der Flashcards-Anwendung vorgenommen werden. Diese Funktion soll leicht erreichbar sein.

Das Erstellen und Löschen von Flashcards soll rückgängig gemacht werden können. Jede Editierung der Seiten einer Flashcard soll rückgängig gemacht werden können. Die Anwendung soll es Erlauben beliebig viele Änderungen rückgängig zu machen. Desweitern soll jede rückgängig gemachte Änderung auch wieder hergestellt werden können.

Aufgabe 2 (3 Punkte)

Ziel: *Implementierung einer User-Story (eines Spike) zur Bestimmung der velocity*

a) Implementieren der User-Story (2,5 Punkte)

Implementieren Sie die in Aufgabe 1 aufgeführte User-Story 1 in Ihrer Flashcards-Anwendung. Nutzen Sie können die Klasse `javax.swing.Timer` nutzen, um ein regelmäßiges Event zu erhalten, dass den Sekundenzähler der Stoppuhr simuliert.

Geben Sie außerdem einen Testplan an, der beschreibt, wie die Korrektheit des Timers mit manuellen Tests sichergestellt werden kann.

b) Verlocity errechenen (0,5 Punkte)

Erarbeiten Sie eine verbesserte Schätzung der User-Stories und bestimmen Sie Ihre Velocity Aufgrund dieser Schätzung.

Aufgabe 3 (4 Punkte)

Ziel: *Erarbeiten einer User-Story und Verbesserung der Schätzung der velocity*

In der Vorstellung dieser Übung wurde eine Anforderung mündlich im Plenum erarbeitet.

a) User-Story erfassen und schätzen (1,5 Punkt)

Erfassen Sie die Anforderung schriftlich als User-Story für Ihr Team.

Schätzen Sie die Komplexität der User-Story und vergeben Sie Story-Points. Begründen Sie Ihre Schätzung kurz.

b) Implementieren der User-Story (2,5 Punkte)

Implementieren Sie die User-Story in Ihrer Flashcards-Anwendung. Geben Sie JUnit Tests für die neue Funktionalität an, die wichtige Fälle testen.